D2.2 Workflow Models Prototype

Description of Software Release

Work package:	**2 Middleware Requirements from Workflows**	
Author(s):	Salem El Sayed, Dirk Pleiter, Simon Smart, Domokos Sarmany, Francois Tessier, Julien Capul	
Reviewer #1	Utz-Uwe Haus	Cray
Reviewer #2	Manuel Arenaz	Appentra
Dissemination Level	Public	
Nature		

Date	Author	Comments	Version	Status
11.07.2019	Salem El Sayed	Input requirements	0.1	Draft
18.07.2019	Salem El Sayed	Restructured and improved defined terms	0.2	Draft
24.07.2019	Simon Smart	Edited front matter	0.3	Draft
28.07.2019	Salem El Sayed	Finalizations	0.4	Draft
30.07.2019	Simon Smart	Proof reading	0.5	Draft
30.08.2019	Utz Haus, Dirk Pleiter	Final edition	1.0	Final

DATA ORCHESTRATION IN HIGH PERFORMANCE COMPUTING
This project has received funding from the European Union's Horizon 2020 research and Innovation program through grant agreement 801101.

Contents

Executive Summery

The MAESTRO project is oriented around a co-design process between application authors along with system and middleware designers. One of the challenges of a co-design process is the inherent complexity of real-world applications and the effort required to port them to different systems or to use newly created functionality. These applications also exhibit large variation of input data and chosen parameters.

As a result, the project partners have built simplified workflows that demonstrate the relevant features of larger processes, along with the configuration and input data to run them in a self-contained manner.

This document is a summary of the workflow prototypes released corresponding to the usage scenarios described in deliverable D2.1.

1. Introduction

Fully featured, operational, HPC workflows are large and unwieldy. The challenges they face at scale motivate the efforts of the project to develop the MAESTRO middleware. However, the effort required to move and restructure these workflows to use a novel middleware, and to run on different systems can be very large.

Within the MAESTRO project, the application partners (ECMWF, JUELICH, CEA and ETHZ) have built stripped-down model workflows. These correspond to the usage scenarios and use cases as described in deliverable D2.1. These workflows can be more straightforwardly modified to make use of the middleware and to demonstrate the impact of the developments in the project.

This deliverable presents the current state of the model workflows. In this context, the term workload includes not only the software, but the parameters configurations and data required to run a body of work on the system. Each of the supplied workflow prototypes is thus a self-contained collection of components, parameters and input data. Further they contain documentation of prerequisites and instructions for building, installing and running the workloads.

The focus of these workflows is on features and functionality relevant to the middleware developments in the MAESTRO project. i.e. a workflow prototype needs to be sufficient to form a demonstrator for the MAESTRO middleware for given usage scenarios and use cases. These workflows will be used as a proof-of-concept for porting applications to the MAESTRO middleware. However, these workflows are not intended to be full operational workflows containing all components, and those which are too complex, proprietary or which detract from the feature demonstration are omitted. Any additional components that have not been supplied will be described in the documentation.

The workflow prototypes included here are snapshots of the associated software. Changes and updates to the software will be communicated to the project partners and be included in deliverable D2.4 to support middleware evaluation.

This deliverable contains the following sections:

- **Section 2** documents the structure of the released workflow prototypes
- **Section 3** gives details of each released workflow prototype.

2. Content Structure

Each workflow prototype is provided in a repository containing the following parts:

Section	Directory and files	Description
Source code	src/<Version> src/Readme	Includes all necessary components to compile and run the given workloads. Ex. src/original src/maestro
Workload	workload/<Size/Name> workload/<Size/Name>/Readme	Required input data and configuration parameters to run the workload. Ex. workload/large workload/medium workload/small
Documentation	doc/	Documentation should explain all steps required to compile, deploy and run the given workloads.

Any deviation from this form will be described if needed.

As the MAESTRO system is developed the testing requirements are likely to become more developed. To fulfil these testing needs, we anticipate that further workloads will be added over the lifetime of the project.

3. Workflow Prototypes

The following lists some notes and location of the released workflow prototypes. The description is listed in this section by application partners.

3.1 Description Template

For each workflow prototype in this deliverable a description of the following items is included in this report.

Usage Scenarios	Number and Name as given by D2.1
Covered use cases	Number and Name as given by D2.1 (Note that any number or combination of usage scenarios and use cases can be covered by a single workflow prototype)
Owner	The project partner releasing the software
Version	The software version that is snapshotted in the workflow prototype released as part of this deliverable (Note that this might be several entries for each software element)
Workload sizes	Listing the available workloads that are part of the workflow prototype released in this deliverable (e.g., Small, Medium, Large)
Release location	Path to finding the released workflow prototype (e.g., URL)
Repository type	Type of code repository used for workflow prototype release (e.g., Git, SVN or Tarball)

Each workflow prototype can be accompanied by further descriptions and details. For example, a brief description can be added for missing components and how their functionality is achieved for the covered use cases.

The main items will be tabulated using the following template:

Usage Scenario	• <Number> <Name> • …
Covered use cases	• <Number> <Name> • …
Owner	
Version	
Workload sizes	
Release location	
Repository type	Git/SVN/Tarball/…

3.2 ECMWF

ECMWF have built a model workflow which captures the essence of the I/O related challenges being tackled within the MAESTRO project. This avoids the large number of additional tasks required to coordinate a full forecast, and simplifies the workflow to its essential components: (mock) data generation, metadata manipulation and data transport and usage.

The workflow is comprised of a few components:

- **Kronos** - Amongst other things, Kronos is an event-driven workload manager. Given a (configurable) schedule, it acts as a meta-scheduler, submitting tasks to the system scheduler. These tasks can notify Kronos of events (startup, completion, error and the completion of intermediate steps) that will in turn trigger further submission of jobs. This component acts as a substitute for our large-scale forecast workflow manager, ecflow.

- **FDB** - The Fields DataBase (FDB) is the library and data service used by ECMWF to handle, transfer, store and index meteorological objects in the forecast pipeline. It provides the I/O layer used by other components. This is the component that we intend to bypass by making use of the MAESTRO middleware.

- **multio-hammer** - The real forecast model is large, uses significant computational resources, is complex to configure and needs to be carefully tuned to specific HPC systems. For the purposes of the MAESTRO project the primary area of interest is the output from the model via its I/O servers. multio-hammer simulates this output process, taking sample data as an input and permuting through the range of metadata required for a forecast run outputting dummy data with various sets of metadata. This data is output through the full I/O stack used in normal operations. In this way it simulates the forecast model from an I/O perspective.

- **pgen** - The post-processing engine used at ECMWF to transform forecast output data into the form requested by customers according to requirement definitions. One pgen task is launched per completed forecast step, and it reads output data from across all ensemble members. The pgen tasks included in this workflow perform tasks reasonably close to the real post-processing, although on synthetic data output from multio-hammer.

Details on how to configure and run this workflow are included in the README in the release repository. It is worth noting that the schedule generator is configured according to the number of forecast steps and forecast ensemble members should be emulated. This allows the workflow to be scaled from very small to (at least) the full size of the operational workload.

Usage Scenario	• US1.1 Operational weather forecast
Covered use cases	• UC1.1 Access semantically-related datasets • UC1.2 High-velocity production of meteorogical objects • UC1.6 Sustained high-volume production of meteorological objects • UC1.7 Consumer applications request sets of objects
Owner	ECMWF
Version	• Kronos: 0.6.0 • atlas: 0.17.2 • ecbuild: 2.10.2 • eckit: 1.0.3 • fdb5: 5.3.5 • metkit: 1.1.0 • mir: 1.2.6 • multio: 0.7.1 • pgen: 1.2.1
Workload sizes	• Variable and configurable. Small to large.
Release location	https://gitlab.version.fz-juelich.de/maestro/ecmwf
Repository type	Git

3.3 Juelich

As described in D2.1, TerrySysMP couples three model components COSMO, CLM and ParFlow. Both COSMO and ParFlow couple to CLM, but not to each other. As a result COSMO (which has a more restrictive license) can be omitted while maintaining the characteristic behaviour relevant to the project. Within the MAESTRO project, the relevant changes will be implemented as part of the coupling between ParFlow and CLM. It is anticipated that the improvements to the two-way coupling can be applied to the coupling of the three models.

In this workflow prototype we have additionally provided a data assimilation tool called PDAF-D (Parallel Data Assimilation Framework), which is used to ingest the small workload provided.

Usage Scenario	• US4.1 Global earth modelling
Covered use cases	• UC4.1 Exchange 2D fields between climate models • UC4.2 Dynamic load balancing of coupled models
Owner	JUELICH
Version	• TerrSysMP: 1.0 • ParFlow: 3.1 • Oasis3-MCT: oasis3-mct_121022 • CLM: 3.5

	• PDAF-D: 1.10
Workload sizes	• Small
Release location	https://gitlab.version.fz-juelich.de/maestro/terrsysmp
Repository type	Git

3.4 CEA

The CEA workflow that will be eventually provided for MAESTRO is representative of a typical chaining scenario of two simulation codes. Chaining of simulation codes is often used to model different physics and/or scales, with the output of the first code being used as the input of the second with an intermediate data transformation/preparation step and a final post-processing step.

Due to the nature of CEA's activities, our actual simulation codes cannot be publicly disseminated. As a substitute we will use the open source proxy application Hydro, a 2D hydrodynamic simulation code, to model our two simulation codes.

Since some development is required to chain Hydro with itself (based on a restart with a different domain decomposition), the workflow prototype currently provided is a simple pipeline consisting of one run of Hydro, following by a post-processing step based on ParaView.

Hydro output data are post-processed with ParaView in batch mode (a python script run with pvpython) to produce a series of images that are compiled into a movie with FFmpeg.

Usage Scenario	• US2.1 Multi-physics simulation pipeline (partly)
Covered use cases	• UC2.1 Applications write/read data collections to/from MAESTRO • UC2.2 "Streaming" data records between producers and consumers • UC2.4 Simulation checkpoints spooling
Owner	CEA
Version	• Hydro (develop branch) • ParaView 5.6.0 • FFmpeg
Workload sizes	• Micro • Small
Release location	https://gitlab.version.fz-juelich.de/maestro/hydro
Repository type	Git

3.5 ETHZ

ETHZ/CSCS have developed SIRIUS, a domain specific library for electronic structure codes. We propose two use cases highlighting the behaviour relevant to the MAESTRO project, that is moving data from main memory to GPU high-bandwidth memory. Both use cases are available in the repository and have variable workload sizes.

Details on how to build SIRIUS and all its dependencies are included in the repository. Documentation of the library, its architecture and our objectives is also available.

Usage Scenario	• US3.1 Electronic structure calculation
Covered use cases	• UC3.1 Move data from CPU to GPU memory • UC3.2 Management of memory resources
Owner	ETHZ
Version	• 6.1.5
Workload sizes	• Variable
Release location	https://gitlab.version.fz-juelich.de/maestro/sirius/
Repository type	Git

4. Concluding remarks

This document is a summary of the workflow prototypes released corresponding to the usage scenarios described in deliverable D2.1. These will be used by the project partners to investigate data use patterns, inform the development of new middleware functionality and provide a baseline against which the new developments can be tested. The workflow prototypes included here are snapshots of the associated software and will be extended and updated through the project. Changes and updates to the software will be communicated to the project partners and be included in deliverable D2.4 to support middleware evaluation.